DOCUMENT RESUME

ED 161 672 SE 024 944

AUTHOR	Weaver, J. F.
TITLE	Calculators and Unary Operations: A Position Paper with Instructional Suggestions. Project Paper 77-7.
INSTITUTION	Wisconsin Univ., Madison. Research and Development Center for Cognitive Learning.
SPONS AGENCY	National Inst. of Education (DHEW), Washington, D.C.
REPORT NO	WRDCCL-PP-77-7
PUB DATE	Dec 77
CONTRACT	NE-C-00-3-0065
NOTE	22p.; Report from the Studies in Mathematics Work Group; For related documents, see SE 024 942-943
EDRS PRICE	MF-$0.83 HC-$1.67 Plus Postage.
DESCRIPTORS	*Algorithms; Calculation; Discovery Learning; *Educational Research; *Instruction; Learning Activities; *Mathematics; Secondary Education; *Secondary School Mathematics
IDENTIFIERS	*Calculators; Computation; Functions; *Number Operations

ABSTRACT

Investigating properties and characteristics of unary operations is shown to be facilitated by using calculators. Several avenues for instructional exploration are suggested and examples given. These include questions relating to relations, inverses, commutativity, distributivity, and iterations. A discussion of the calculator algorithm for exploring each question is included. (MP)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED161672

Project Paper 77-7

CALCULATORS AND UNARY OPERATIONS

(A Position Paper, with instructional suggestions)

by

J. F. Weaver

U S DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

J. F. Weaver

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC) AND USERS OF THE ERIC SYSTEM."

Report from the
Studies in Mathematics Work Group

Thomas A. Romberg
Work Group Chairman

Thomas C. Carpenter, John G. Harvey, and J. Fred Weaver
Faculty Associates

James M. Moser
Senior Scientist

Wisconsin Research and Development Center
for Cognitive Learning
The University of Wisconsin
Madison, Wisconsin

December 1977

SE 024 944

(This paper is being submitted for potential journal publication)

CALCULATORS AND UNARY OPERATIONS

J. F. Weaver
The University of Wisconsin-Madison
Madison, Wisconsin

Several years ago Hess (1974) indicated that from early on, school mathematics programs and texts give much emphasis to binary operations (diadic functions) but relatively little attention to unary operations (monadic functions), which may unwittingly prompt students to conclude that binary operations "are the only important operations in mathematics [p. 281]." This condition may have changed somewhat within the past few years, but not markedly so.

Unary operations on numbers are encountered in a variety of calculations. More important mathematically, however, is the fact that unary operations have properties and distinguishing characteristics just as do binary operations.

Systematic consideration of selected unary operations can be facilitated by using electronic calculators. It is the intent of this paper to suggest and illustrate some of the ways in which this can be accomplished. Emphasis is placed upon unary operations that are particularly suitable for exploration and investigation at a pre-algebra level, using *calculator algorithms* as the basis for generating examples or nonexamples to develop or test conjectures.

Domain of Definition

In a formal sense, as Hess (1974) has indicated, "Unary operations and binary operations are particular cases of n-ary operations [p. 281]." A student may be well along in the study of mathematics, however, before comprehending the preceding simply-phrased statement in these terms (Lay, 1966):

> "Let us now consider a rule of correspondence that assigns to *ordered subsets of n elements* of S a unique element of T. This will be called an *n*-ary operation from the set S into the set T. If this correspondence applies to every possible ordered subset of *n* elements in S, the operation is said to be *complete*, or *performable* on S. If the operation is complete and T = S, then the operation is closed on S. Since both the domain of definition and the range of images are now in S, we say we have an *n*-ary operation in S. According to whether $n = 1, 2, 3, \ldots, n$ the operation is said to be unary, binary, ternary, . . ., *n*-ary [p. 198]."

At times, especially at a pre-algebra level, it is important and necessary to distinguish between what I shall call the *canonical* and the *expedient* domains of a unary operation. The former refers to the domain over which a particular operation customarily is defined within the algebra of the real number system; the latter, to a domain that (for one reason or another) is a proper subset of the canonical domain. For instance:

The *squaring* operation, $a \rightarrow a^2$, has as its canonical domain the set of real numbers. However, at a particular instructional level this unary operation may have for its expedient domain the set of non-negative integers, or the set of rational numbers, etc.,--simply because at a given instructional level students' knowledge of number does not go beyond some distinguished proper subset of the reals.

(There are some unary functions, of course, for which the canonical domain itself is some proper subset of the reals; e.g., $a \rightarrow \log a$.)

As teachers use any of the instructional suggestions given in later sections of this paper, they will have to decide which domains--canonical or particular expedient ones--are appropriate within their own classroom situations.

Notes on Notation

Unary operations or monadic functions are customarily symbolized in a variety of ways, with different notations used at times for the same operation or function. For instance:

$\log x$	$x!$
$\sqrt{x}$	x^2
$\lvert x \rvert$	x^{-1} or $\frac{1}{x}$ or $1/x$
$[x]$	e^x

As Gilman & Rose (1976) have indicated, "Whatever other merits this mishmash has, consistency certainly isn't one of them, for the symbol which is the functional indicator may appear on the left, the right, both sides, on top, or be in a special position, or be represented by an alphabetical label [p. 57]!"

Such inconsistency is intolerable in a computer language, among other things. One way to resolve the situation is to develop a special set of symbols so that unary or monadic functions may be indicated consistently in a form such as "$\nabla\ a$" which is termed *prefix* or *pre-operator* or *left-operator* notation. The unary operator (∇), however it may be symbolized, is written immediately *before* the argument or operand (a). This is the resolution adopted by *APL* (*A Programming Language*), for instance.

Another way to resolve the situation is to use a notation of the form "$a\ \nabla$" which is termed *postfix* or *post-operator* or *right-operator* notation in which the unary operator (∇), however it may be symbolized, is written immediately *after* the argument or operand (a). With minor modification this is the notational form used in this paper and is judged to be the preferable one for a *calculator* language, regardless of whether the calculator procedures for data entry and processing are based upon "algebraic logic" (without or with an operational hierarchy) or upon RPN (reverse Polish notation). This does not

resolve *all* of the notational issues identified in the *Report of the conference on needed research and development on hand-held calculators in school mathematics* [n.d., p. 32], but it does provide a notational consistency across calculators when working with unary or monadic operations or functions.

These are symbolized throughout this paper in the general form "a [∇]" where a is the argument or operand and [∇] is a facsimile, as it were, of a particular calculator key, with ∇ replaced by the symbol on the face of that key. This is consistent with the fact that calculator-computation of the value of a particular monadic function applied to a given argument is effected by *first* entering and/or displaying the operand (a) and *then* activating the appropriate operator key ([∇]).

In this paper emphasis is given to four monadic functions, each with the indicated notation for *calculator execution* of that function:

1. The *squaring* function (i.e., $a \rightarrow a^2$) ..... a [x^2]
2. The *principal square root* function (i.e., $a \rightarrow \sqrt{a}$) .... a [$\sqrt{x}$]
3. The *additive inverse* function (i.e., $a \rightarrow {}^{-}a$) ..... a [CHS]
4. The *reciprocal* or *multiplicative inverse* function (i.e., $a \rightarrow \frac{1}{a}$ or $a \rightarrow a^{-1}$) ..... a [1/x]

(Although some calculators have a [+/-] key rather than a [CHS] key for function 3, I personally prefer the latter notation for the intended conceptualization. It also is recognized that in some of the "early" electronic calculators such as Hewlett-Packard's model HP-35, the [CHS] key or its equivalent could be executed as either a pre- or post-operator; or in the case of some calculators, as a pre- rather than post-operator. To the best of my knowledge current models of calculators use virtually exclusively a post-operator execution position for this additive inverse function.)

Relatively less emphasis is given in this paper to the following monadic functions:

5. The *common logarithm* function (i.e., $a \rightarrow \log a$) ..... a [log]

6. The *common antilog* function (i.e., $a \rightarrow 10^a$) ..... a [10^x]

(Function 6 assumes a calculator with a hard-wired [10^x] key. If execution had to be effected using a [y^x] key in conjunction with keyboard entry of both a and 10, some persons would contend that now we are dealing with a *diadic* or *binary* operation rather than a monadic one.)

Work with the first four of the preceding unary operators is quite appropriate for, and has been carried out with, pre-algebra students at the middle or junior high school level.[1] Work with the latter two unary operators generally is more suitable in connection with algebra rather than pre-algebra courses, although it certainly is not ruled out for the latter.

Notation-related Matters

Two notation-related matters should be identified and discussed briefly.

1. The *calculator notation* adopted in this paper for unary operations is intended to mirror the keystroke sequences used in evaluating exemplars of such functions. In the case of some calculators, however, a "shift" or "prefix" key must be used prior to an operator key in order to select the desired operator associated with a two- or three-function key. Such a "shift" or "prefix" key is *not* incorporated in the *notation* used in this paper, however.

2. The problems associated with interpreting an expression such as "-(-5)," which Musser (1972) has discussed at some length and which Van Engen (1972) has termed "just a fact of life for a mathematics teacher [p. 299]," are not eliminated when using a calculator; in fact, they may be magnified by use of the [CHS] key. For instance, in order to evaluate "-5[CHS]," the following *keystroke sequence* is used: 5[CHS][CHS], in which the two [CHS] keystrokes accomplish *different* things *conceptually*.

It is important to distinguish between the [CHS] *operator* applied to operand a, and an *implicit* [CHS] calculator *keystroke* executed to input an operand $a < 0$, to which [CHS] or some other operator is then applied.

Instructional Suggestions

The material that follows is intended simply to *suggest* avenues for instructional exploration and investigation. It is assumed that teachers will exercise considerable latitude in the explicit ways in which the suggestions may be incorporated into ongoing instructional programs.

In any event, one thing in particular should be kept in mind: Since calculators in many instances generate and display *rational approximations* to real numbers rather than real numbers *per se*, this condition must be handled accordingly when attempting to ascertain the validity of conjectured properties from specific numerical exemplars. The consequences of "rounding" or "truncating" must be recognized and taken into account when necessary.

Graphs

The construction of a portion of a graph of each of the following functions (assuming a to be a real number) can be helpful in getting an initial "feel" for the unary operators involved--their similarities and their differences--and also in dealing with some subsequent considerations:

1. $a \rightarrow a\boxed{x^2}$
2. $a \rightarrow a\boxed{\sqrt{x}}$
3. $a \rightarrow a\boxed{\text{CHS}}$
4. $a \rightarrow a\boxed{1/x}$

It will be noted from Figures 1 and 2, for instance, that the partial graphs for the latter two of the preceding functions are markedly different from each other. How do the graphs of the first two of the preceding functions compare with each other? with Figures 1 and 2?

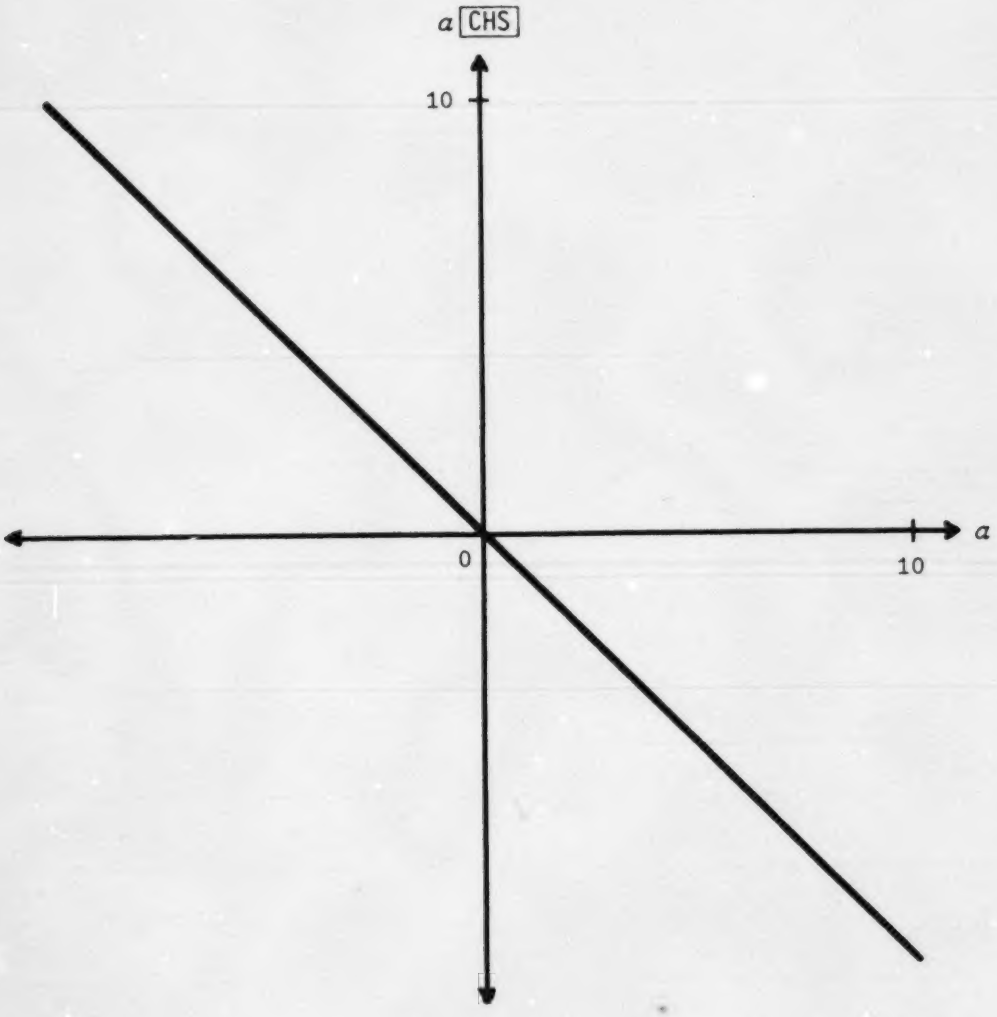

Figure 1. Partial graph of $a \rightarrow a$ CHS

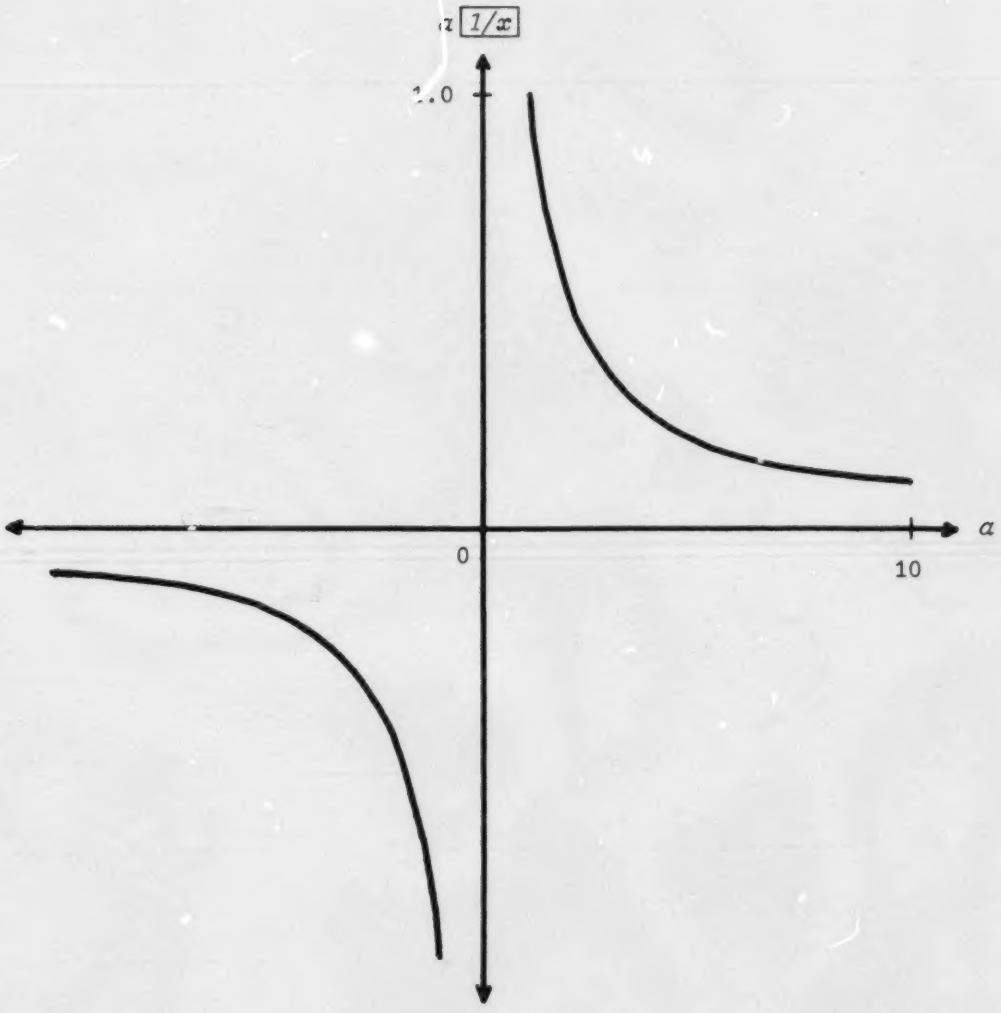

Figure 2. Partial graph of $a \to a\boxed{1/x}$

Relations

Some unary operations maintain the sense of certain binary relations. Other unary operations do not.

For each of the following: For all a,b that make the antecedent true, is the consequent also true? If not, under what condition(s) would it be true?

1. $a = b \Rightarrow a\boxed{x^2} = b\boxed{x^2}$
2. $a > b \Rightarrow a\boxed{x^2} > b\boxed{x^2}$
3. $a < b \Rightarrow a\boxed{x^2} < b\boxed{x^2}$
4. $a = b \Rightarrow a\boxed{\sqrt{x}} = b\boxed{\sqrt{x}}$
5. $a > b \Rightarrow a\boxed{\sqrt{x}} > b\boxed{\sqrt{x}}$
6. $a < b \Rightarrow a\boxed{\sqrt{x}} < b\boxed{\sqrt{x}}$
7. $a = b \Rightarrow a\boxed{\text{CHS}} = b\boxed{\text{CHS}}$
8. $a > b \Rightarrow a\boxed{\text{CHS}} > b\boxed{\text{CHS}}$
9. $a < b \Rightarrow a\boxed{\text{CHS}} < b\boxed{\text{CHS}}$
10. $a = b \Rightarrow a\boxed{1/x} = b\boxed{1/x}$
11. $a > b \Rightarrow a\boxed{1/x} > b\boxed{1/x}$
12. $a < b \Rightarrow a\boxed{1/x} < b\boxed{1/x}$

Inverses

Some unary operations are inverses of each other. Certain unary operations are their own inverses.

For each of the following: Is the sentence true for all a ? If not, under what condition(s) is the sentence true?

1. $a\boxed{x^2}\boxed{\sqrt{x}} = a$
2. $a\boxed{\sqrt{x}}\boxed{x^2} = a$
3. $a\boxed{\text{CHS}}\boxed{\text{CHS}} = a$
4. $a\boxed{1/x}\boxed{1/x} = a$

Commutativity

For each of the following: Is the sentence true for all a ? If not, under what condition(s) is it true?

1. $a\,[x^2]\,[\text{CHS}] = a\,[\text{CHS}]\,[x^2]$
2. $a\,[x^2]\,[1/x] = a\,[1/x]\,[x^2]$
3. $a\,[\sqrt{x}]\,[\text{CHS}] = a\,[\text{CHS}]\,[\sqrt{x}]$
4. $a\,[\sqrt{x}]\,[1/x] = a\,[1/x]\,[\sqrt{x}]$
5. $a\,[\text{CHS}]\,[1/x] = a\,[1/x]\,[\text{CHS}]$

Distributivity

For each of the following: Is the sentence true for all a,b ? If not, under what condition(s) is it true?

1. $(a + b)\,[x^2] = a\,[x^2] + b\,[x^2]$
2. $(a - b)\,[x^2] = a\,[x^2] - b\,[x^2]$
3. $(a \times b)\,[x^2] = a\,[x^2] \times b\,[x^2]$
4. $(a \div b)\,[x^2] = a\,[x^2] \div b\,[x^2]$
5. $(a + b)\,[\sqrt{x}] = a\,[\sqrt{x}] + b\,[\sqrt{x}]$
6. $(a - b)\,[\sqrt{x}] = a\,[\sqrt{x}] - b\,[\sqrt{x}]$
7. $(a \times b)\,[\sqrt{x}] = a\,[\sqrt{x}] \times b\,[\sqrt{x}]$
8. $(a \div b)\,[\sqrt{x}] = a\,[\sqrt{x}] \div b\,[\sqrt{x}]$
9. $(a + b)\,[\text{CHS}] = a\,[\text{CHS}] + b\,[\text{CHS}]$
10. $(a - b)\,[\text{CHS}] = a\,[\text{CHS}] - b\,[\text{CHS}]$
11. $(a \times b)\,[\text{CHS}] = a\,[\text{CHS}] \times b\,[\text{CHS}]$
12. $(a \div b)\,[\text{CHS}] = a\,[\text{CHS}] \div b\,[\text{CHS}]$
13. $(a + b)\,[1/x] = a\,[1/x] + b\,[1/x]$
14. $(a - b)\,[1/x] = a\,[1/x] - b\,[1/x]$
15. $(a \times b)\,[1/x] = a\,[1/x] \times b\,[1/x]$
16. $(a \div b)\,[1/x] = a\,[1/x] \div b\,[1/x]$

Exemplars for each of the preceding 16 sentence types may be generated from a *calculator algorithm* as suggested by Table 1 or Table 2 (depending upon the kind of calculator used), where [○] and [▽] may be replaced by some particular binary and unary operator respectively. In each such instance, is $c = d$ for all a,b ? (Or, the algorithms could be extended very readily to answer the equivalent question: Is $c - d = 0$?

TABLE 1

An Algebraic-logic Calculator Algorithm for Distributivity Sentences

Left-hand member				Right-hand member			
INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS	INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS
a	○			a	▽	○	
b	=	▽	c	b	▽	=	d

TABLE 2

An RPN Calculator Algorithm for Distributivity Sentences

Left-hand member				Right-hand member			
INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS	INPUT DATA/UNITS	KEYS		OUTPUT DATA/UNITS
a	ENTER↑			a	▽		
b	○	▽	c	b	▽	○	d

Iterations

An iteration of the general form

a [▽] [▽] [▽] [▽] [▽] . . .

may be studied by generating, term by term, the following sequence

$n_1 = a_i$ [▽]

$n_2 = a_i$ [▽] [▽]

$n_3 = a_i$ [▽] [▽] [▽]

$n_4 = a_i$ [▽] [▽] [▽] [▽]

$n_5 = a_i$ [▽] [▽] [▽] [▽] [▽]

etc.

for some particular [▽] and a_i, and then across sequences generated from the same [▽] applied to various a_i's.

In what way(s), if at all, do iterations 1 and 2 below "behave" more or less similarly for all a ?

1. a [CHS] [CHS] [CHS] [CHS] [CHS] . . .
2. a [1/x] [1/x] [1/x] [1/x] [1/x] . . .

Can the following iteration 3 be characterized in the same way for all a ?

3. a [x^2] [x^2] [x^2] [x^2] [x^2] . . .

In what way(s) is iteration 4 below for all $0 < a < 1$ similar to that for all $a > 1$?

4. a [$\sqrt{x}$] [$\sqrt{x}$] [$\sqrt{x}$] [$\sqrt{x}$] [$\sqrt{x}$] . . .

Although a bit more complex in its structure, a much more mathematically interesting iteration involving the [$\sqrt{x}$] operator may be generated progressively in the following manner:

5. $n_1 = a\boxed{\sqrt{x}}$

$n_2 = (a\boxed{\sqrt{x}} + a)\boxed{\sqrt{x}}$

$n_3 = ((a\boxed{\sqrt{x}} + a)\boxed{\sqrt{x}} + a)\boxed{\sqrt{x}}$

$n_4 = (((a\boxed{\sqrt{x}} + a)\boxed{\sqrt{x}} + a)\boxed{\sqrt{x}} + a)\boxed{\sqrt{x}}$

etc.

That is to say, we are dealing with:

$$n = \sqrt{a + \sqrt{a + \sqrt{a + \sqrt{a + \cdots}}}}$$

For a given a, convergence to *some* number n is relatively rapid. It might be anticipated that n is "necessarily" irrational. But such is *not* "necessarily" the case at all. With a suitable choice for a, n can be made to approach an *integer;* not just *some* integer, but *any desired integer greater than 1.* It can be shown that $a = n(n - 1)$ [Olgivy & Anderson, 1966, p. 62]. So, pick your integer $n > 1$; then calculate the a to be used.

For instance: Convergence to $n = 3$ can be effected by using $a = 3(3 - 1) = 6$. This is illustrated by Table 3 which, among other things, translates the iterative procedure into keystrokes for representative kinds of calculators.[2]

Extensions

Further explorations can be made, of course, with the unary operators considered in this paper. Also, other unary operators can be introduced and investigated: e.g., the (common) $\boxed{\log}$ operator, and its inverse $\boxed{10^x}$.

Given each of the following, for instance, under what condition(s)--if any--is it valid for all a, or for all b ?

TABLE 3

Exemplar for Iterations: 5

TI-1270 or TI-30 keystrokes	HP-21 keystrokes	Truncated display	Iteration approximated
6 [STO]	6 [ENTER↑] [ENTER↑] [ENTER↑]	6.00000	
[√x]	■ [√x]	2.44948	$n_1 = \sqrt{6}$
[+] [RCL] [=] [√x]	[+] ■ [√x]	2.90680	$n_2 = \sqrt{6 + \sqrt{6}}$
[+] [RCL] [=] [√x]	[+] ■ [√x]	2.98442	$n_3 = \sqrt{6 + \sqrt{6 + \sqrt{6}}}$
[+] [RCL] [=] [√x]	[+] ■ [√x]	2.99740	$n_4 = \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6}}}}$
[+] [RCL] [=] [√x]	[+] ■ [√x]	2.99956	$n_5 = \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6}}}}}$
[+] [RCL] [=] [√x]	[+] ■ [√x]	2.99992	$n_6 = \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6}}}}}}$
[+] [RCL] [=] [√x]	[+] ■ [√x]	2.99998	$n_7 = \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6}}}}}}}$
[+] [RCL] [=] [√x]	[+] ■ [√x]	2.99999	$n_8 = \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6 + \sqrt{6}}}}}}}}$

1. a [log] [10^x] = a
2. a [10^x] [log] = a
3. $(a + b)$ [log] = a [log] + b [log]
4. $(a - b)$ [log] = a [log] - b [log]
5. $(a \times b)$ [log] = a [log] × b [log]
6. $(a \div b)$ [log] = a [log] ÷ b [log]

The following attempted iteration leads to consequences that often are not immediately evident to students at the outset:

7. a [log] [log] [log] [log] [log] · · ·

Knowing a, can the maximum number of iterations be predicted?

Can valid iterations of the following nature be generated for $a < 0$?

8. a [10^x] [10^x] [10^x] [10^x] [10^x] · · ·

In Conclusion

Unary operations are much more than things to be executed as occasion demands in the course of computation or calculation. Such operations have distinguishing properties just as do binary operations. Using calculators in the ways suggested or implied in this paper can help students view unary operations in appropriate perspectives, with attention to their significant characteristics as mathematical functions.

In connection with calculator algorithms, some form of *post*fix notation (rather than *pre*fix notation) is deemed advisable for unary operators since such notation reflects the point of operator execution in a keystroke sequence. Consider, for instance, evaluation of an expression of the form $\sqrt{a^2 + b^2}$.

Algebraic-logic calculator: a [x^2] [+] b [x^2] [=] [$\sqrt{x}$].

RPN calculator: a [x^2] b [x^2] [+] [$\sqrt{x}$].

F[illegible]y, it should be noted in the preceding algorithms that although an alg[illegible]raic-logic calculator algorithm uses postfix notation for unary operators, a "*between*fix" notation is used for *binary* operators. In the case of an RPN calculator algorithm, however, postfix notation is used for unary and binary operators alike.

This latter observation often is viewed as an advantage for RPN calculators and makes possible the simple flowchart shown in Figure 3 (*HP-35 operating manual* [p. 34], 1973), or the Figure 4 modification that I have used in connection with classroom instruction[3] involving work with RPN calculators by accelerated seventh-grade students in a pre-algebra course.[1]

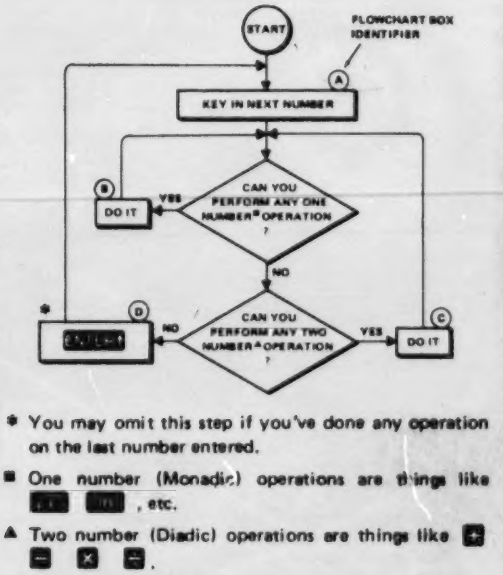

Figure 3. An H-P flowchart to evaluate numerical expressions

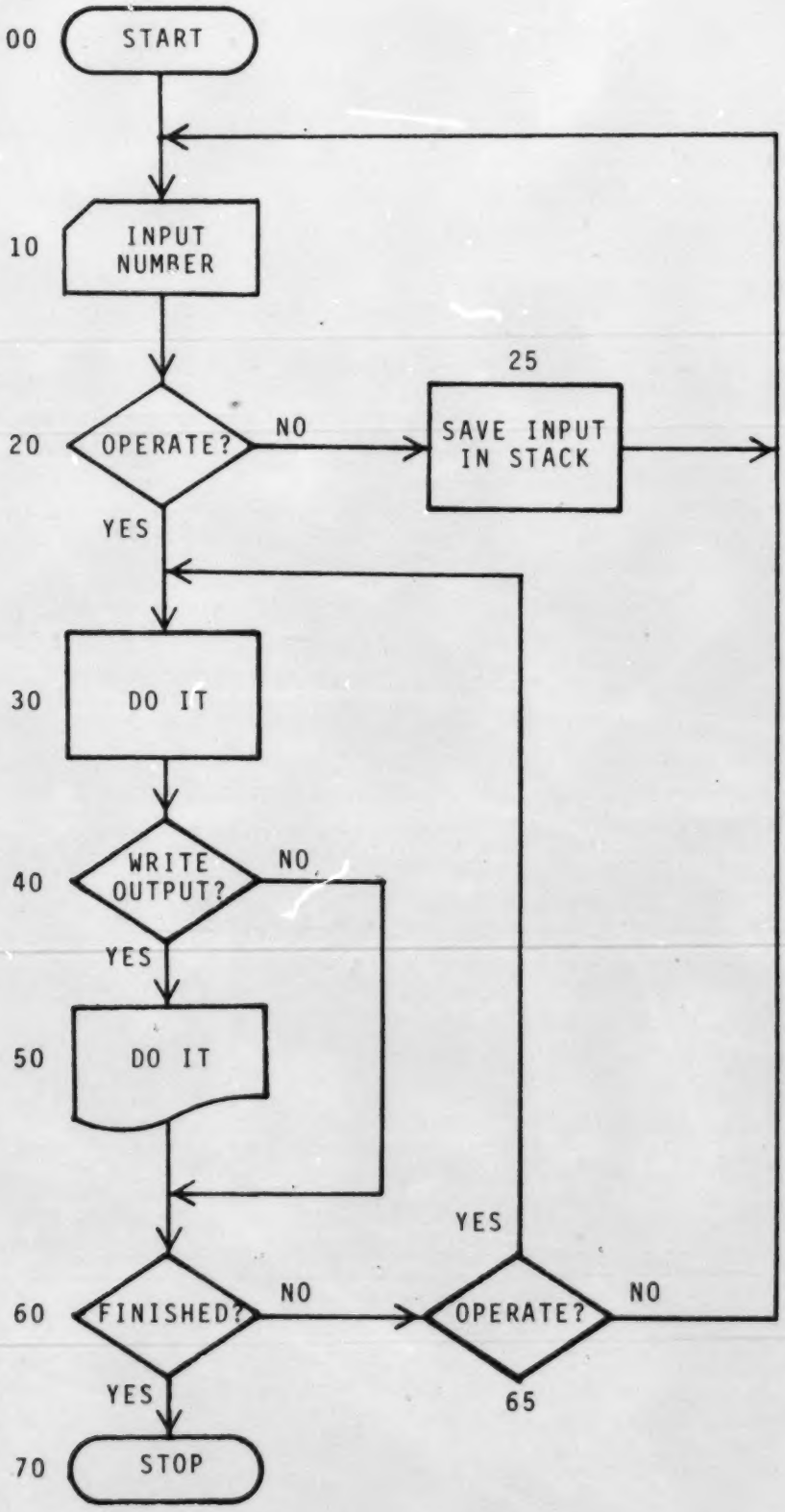

Figure 4. A modification of Figure 3

References

Gilman, L., & Rose, A. J. *APL, An interactive approach* (2nd ed., rev. reprint.). New York: John Wiley & Sons, 1976.

Hess, A. L. Unary operations. *Mathematics Teacher,* 1974, *67,* 281-283.

HP-35 operating manual. Cupertino, CA: Hewlett-Packard, 1973.

Lay, L. C. *The study of arithmetic.* New York: Macmillan, 1966.

Musser, G. L. Additive inverses and negatives--Two different concepts. *Mathematics Teacher,* 1972, *65,* 301-304.

Ogilvy, C. S., & Anderson, J. T. *Excursions in number theory.* New York: Oxford University Press, 1966.

Report of the conference on needed research and development on hand-held calculators in school mathematics. Washington, DC: National Institute of Education & National Science Foundation, n.d.

Van Engen, H. 'Tain't necessarily so, or minus minus *A* equals *A*. *Mathematics Teacher,* 1972, *65,* 299-300.

Footnotes

[1] This work was done by the writer in connection with his activities as a Faculty Associate, Studies in Mathematics Work Group, Wisconsin Research and Development Center for Cognitive Learning--supported in part as a research and development center by funds from the National Institute of Education, Department of Health, Education, and Welfare (Center Contract No. NE-C-00-3-0065). The opinions expressed herein do not necessarily reflect the position or policy of the National Institute of Education and no official endorsement by that agency should be inferred.

[2] Two things should be noted about the indicated keystroke sequences: (1) Some algebraic-logic calculators, unlike the TI-1270 and TI-30, have a four-key memory such as: [CM], [MR], [M+] and [M-]. For such calculators, [CM] [M+] would replace [STO], and [MR] would be the same as [RCL] in the specified keystroke sequence. (2) Some RPN calculators such as National Semiconductor's "Mathematician" and "Scientist" models do not permit "stack storage" as does the HP-21, due to a difference in the "behavior" of the top stack register when the stack drops. This can be accommodated, however, in such non-HP calculators by a slightly more involved routine of data manipulations (not detailed here).

[3] Two things should be noted about the flowcharts (Figures 3 and 4). (1) The number of levels or registers in a calculator's operational stack --usually four, sometimes three--places a practical limit upon the number of operands that can be held in storage prior to the execution of a unary or binary operation. The flowcharts, *as drawn*, do not take this limitation into account and more or less imply a stack of "infinitely many" levels or registers. (2)The flowcharts are not intended to cover data storage in *addressable* registers, or other data manipulations--things that may be needed when using a calculator with a 4- or 3-level operational stack, but not with an "idealized" infinitely-many-level stack.